IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Subhashini Subramaniam	Art Unit: 3623
Appl. No.: 10/722,408	Examiner: CHOI, PETER H
Filed: 11/28/2003	Attorney Docket No.: SUN-007/030215
For: Meta Directory Server Providing Users the Ability to Customize Work-flows	

Appeal Brief Under 37 CFR § 41.37

Mail Stop **Appeal Brief - Patents**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the Final Office Action dated 29-Oct-2009 ("Outstanding Final Office Action") and further to the Notice of Appeal filed on 29-JAN-2010, Appellants submit this appeal brief under 37 CFR § 41.37.

As required by 37 C.F.R. § 41.37, this brief contains items under the following headings:

I. REAL PARTY IN INTEREST

II. RELATED APPEALS AND INTERFERENCES

III. STATUS OF CLAIMS

IV. STATUS OF AMENDMENTS

V. SUMMARY OF CLAIMED SUBJECT MATTER

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

VII. THE ARGUMENT

VIII. CLAIMS

IX. EVIDENCE APPENDIX

X. RELATED PROCEEDINGS APPENDIX

I. REAL PARTY IN INTEREST

This application is assigned to Sun Microsystems Inc. (acquired now by Oracle International, Inc.) by virtue of the assignment recorded on 11/28/2003 at Reel/Frame: 014748/0103 at the USPTO.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN THE APPLICATION

There are 29 claims currently pending in the application. Claims 1, 11, 21 and 31 are independent claims.

B. STATUS OF ALL CLAIMS

Claims pending: 1, 3-6, 8-11, 13-16, 18-21, 23-26, 28-31 and 33-36

Claims canceled: 2, 7, 12, 17, 22, 27 and 32

Claims withdrawn: None

Claims allowed: None

Claims objected to: None

Claims rejected: 1, 3-6, 8-11, 13-16, 18-21, 23-26, 28-31 and 33-36

C. CLAIMS ON APPEAL

All the rejected claims 1, 3-6, 8-11, 13-16, 18-21, 23-26, 28-31 and 33-36 are subject of this appeal.

IV. STATUS OF AMENDMENTS

The Appellants have not filed any amendments after the Final Office Action dated 10/29/2009.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present invention is directed to providing users the ability to customize work-flows. Figure 1 (reproduced below) of the subject application shows an example general environment in which the features of the invention can be implemented.

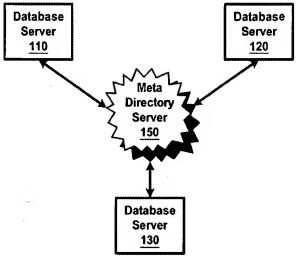

FIG. 1

Independent claim 1 relates to a method of enabling a user to extend a work flow (paragraph 006, Figure 2 and paragraph 0033) for synchronization/consolidation of data (paragraph 004, 006) between at least two data sources (paragraph 004, database servers 110 and 120 of Figure 1 and paragraph 0034). The work flow is recited as being for execution in a meta directory server (150 of Figure 1, paragraphs 004-006, 0031). The method is recited to contain:

providing, by a designer (paragraphs 009, 0078), a sequence of built-in tasks (paragraphs 006, steps 210-250 of Figure 2 and paragraphs 0033, 0035-0040) which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources (paragraph 0033), a built-in task (steps 210, 225, 235, 240 of Figure 2) in said sequence of built-in tasks containing an extension point (paragraphs 008, 0045, 0050) at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances (paragraph 0046),

wherein each user is provided the ability to specify a corresponding custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension

point without editing said work flow, the custom task specified by a user (paragraph 0046) containing corresponding program logic to provide a customization desired by the user (paragraph 0075),

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048);

receiving from said user data indicating a custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user (paragraph 0046) for the corresponding desired customization;

executing (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048) said custom task when said extension point is reached during execution of said built-in task in an instance of said work-flow for said user; and

continuing execution (paragraph 0027, steps 315, 320, 370, 380 of Figure 3 and paragraphs 0047, 0049 and 0050) of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/consolidating data between said two data sources.

Thus, in accordance with claim 1, a designer first specifies extension points at corresponding points of interest in a workflow, and users may thereafter specify corresponding custom tasks at the extension points. Due to such a feature, no user needs to edit the workflow, but can specify corresponding custom tasks of interest at the extension points.

Dependent claim 3 recites that said custom task (paragraph 0010, CA tasks 460, 465 of Figure 4 and paragraph 0054) contains an another extension point. The method is recited to further contain receiving from said user data (paragraph 0010) indicating an another custom task (CA tasks 465, 470 of Figure 4 and paragraph 0054) to be executed when said another extension point is reached during execution of said custom task.

Dependent claim 4 recites that the method further contains:

determining (paragraph 0011, Figure 7 line 760 and associated paragraph 0085) corresponding set of extension points available in each of said sequence of built-in tasks;

displaying (paragraph 0011, 900 of Figure 9 and paragraphs 0084 and 0085) each of said set of extension points (930 of Figure 9) associated with a corresponding one (919 of Figure 9) of said sequence of built-in tasks;

displaying (950 of Figure 9) said custom task and said another custom task; and

enabling (900 of Figure 9 and paragraphs 0086 and 0087) said user to specify said custom task associated with said extension point, and said another custom task associated with said another extension point.

Dependent claim 5 recites enabling said user to specify that said custom task is to be executed synchronously (900 of Figure 9 and paragraph 0088), wherein execution of said sequence of built-in tasks is suspended at said extension point during execution of said custom task, and wherein execution of said sequence of built-in tasks is resumed after completion of execution of said custom task (paragraphs 0012 and 0053) such that said custom task is executed in a synchronous manner.

Dependent claim 6 recites enabling said user to specify that said custom task is to be executed asynchronously(900 of Figure 9 and paragraph 0088), wherein said custom task is executed in parallel with execution of built-in task from said extension point (paragraphs 0012 and 0054) such that said custom task is executed in an asynchronous manner.

Dependent claim 8 recites that at least one of said two data sources comprises a relational database (paragraphs 0004 and 0005).

Independent claim 13 recites a computer readable medium (1040 of Figure 10, paragraph 0098) storing one or more sequences of instructions for causing a meta directory server (150 of Figure 1, paragraphs 004-006, 0031) to enable a user to extend a work flow (paragraph 006, Figure 2 and paragraph 0033) for synchronization/consolidation of data (paragraph 004, 006) between at least two data sources (paragraph 004, database servers 110 and 120 of Figure 1 and paragraph 0034). The execution of said one or more sequences of instructions by one or more processors contained in said meta directory server causes said meta directory server to perform the actions of:

providing, by a designer (paragraphs 009, 0078), a sequence of built-in tasks (paragraphs 006, steps 210-250 of Figure 2 and paragraphs 0033, 0035-0040) which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources (paragraph 0033), a built-in task (steps 210, 225, 235, 240 of Figure 2) in said sequence of built-in tasks containing an extension point (paragraphs 008, 0045, 0050) at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances (paragraph 0046),

wherein each user is provided the ability to specify a corresponding custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point without editing said work flow, the custom task specified by a user (paragraph 0046) containing corresponding program logic to provide a customization desired by the user (paragraph 0075),

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048);

receiving from said user data indicating a custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user (paragraph 0046) for the corresponding desired customization;

executing (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048) said custom task when said extension point is reached during execution of said built-in task in an instance of said work-flow for said user; and

continuing execution (paragraph 0027, steps 315, 320, 370, 380 of Figure 3 and paragraphs 0047, 0049 and 0050) of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/consolidating data between said two data sources.

Independent claim 21 recites a meta directory server (150 of Figure 1, paragraphs 004-006, 0031) enabling a user to extend a work flow (paragraph 006, Figure 2 and paragraph 0033) for synchronization/consolidation of data (paragraph 004, 006) between at

least two data sources (paragraph 004, database servers 110 and 120 of Figure 1 and paragraph 0034). The meta directory server is recited as containing:

means for providing, by a designer (paragraphs 009, 0078), a sequence of built-in tasks (paragraphs 006, steps 210-250 of Figure 2 and paragraphs 0033, 0035-0040) which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources (paragraph 0033), a built-in task (steps 210, 225, 235, 240 of Figure 2) in said sequence of built-in tasks containing an extension point (paragraphs 008, 0045, 0050) at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances (paragraph 0046),

wherein each user is provided the ability to specify a corresponding custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point without editing said work flow, the custom task specified by a user (paragraph 0046) containing corresponding program logic to provide a customization desired by the user (paragraph 0075),

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048);

means for receiving from said user data indicating a custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user (paragraph 0046) for the corresponding desired customization;

means for executing (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048) said custom task when said extension point is reached during execution of said built-in task in an instance of said work-flow for said user; and

means for continuing execution (paragraph 0027, steps 315, 320, 370, 380 of Figure 3 and paragraphs 0047, 0049 and 0050) of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/consolidating data between said two data sources.

Independent claim 31 recites a meta directory server (150 of Figure 1, paragraphs 004-006, 0031) enabling a user to extend a work flow (paragraph 006, Figure 2 and

paragraph 0033) for synchronization/consolidation of data (paragraph 004, 006) between at least two data sources (paragraph 004, database servers 110 and 120 of Figure 1 and paragraph 0034). The meta directory server is recited as containing:

a task registry block (510 of Figure 5, paragraph 0059) storing data related to a sequence of built-in tasks (paragraphs 006, steps 210-250 of Figure 2 and paragraphs 0033, 0035-0040) which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources (paragraph 0033), a built-in task (steps 210, 225, 235, 240 of Figure 2) in said sequence of built-in tasks containing an extension point (paragraphs 008, 0045, 0050) at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances (paragraph 0046),

wherein each user is provided the ability to specify a corresponding custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point without editing said work flow, the custom task specified by a user (paragraph 0046) containing corresponding program logic to provide a customization desired by the user (paragraph 0075),

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048);

a user interface module (530 of Figure 5 and paragraph 0058) receiving from said user data indicating a custom task (paragraph 007, steps 260 of Figure 2 and paragraph 0040) associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user (paragraph 0046) for the corresponding desired customization; and

work-flow manager module (540 of Figure 5 and paragraphs 0065 and 0066) for executing (paragraph 008, steps 320, 330, 340, 350 of Figure 3 and paragraphs 0047 and 0048) said custom task when said extension point is reached during execution of said built-in task, and in addition for continuing execution (paragraph 0027, steps 315, 320, 370, 380 of Figure 3 and paragraphs 0047, 0049 and 0050) of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/ consolidating data between said two data sources.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

(A) Rejection of claims 1, 3-6, 8-11,13-16, 18-21, 23-26, 28-31, and 33-36 under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement.

(B) Rejection of claims 1, 3, 4, 8-11, 13-14, 18-21, 23-24, 28-31 and 32-34 under 35 U.S.C. § 103(a) as being unpatentable over Chatterjee *et al* (US Patent # 5,774,661) in view of Hansen *et al* (US Patent #7,013,316) and Bacon *et al* (US Patent #6,430,538).

(C) Rejection of claims 5-6, 15-16, 25-26, and 35-36 under 35 U.S.C. § 103(a) as being unpatentable over Chatterjee, in view of Hansen and Bacon, as applied to claims 1, 11, 21 and 31 above, and further in view of Randell (US Patent #5,745,687).

VII. THE ARGUMENT

A. Rejection of claims 1, 3-6, 8-11,13-16, 18-21, 23-26, 28-31, and 33-36 under 35 U.S.C. § 112, first paragraph

Claims 1, 11, 21, and 31 were rejected under 35 U.S.C. § 112, allegedly as failing to comply with the written description requirement. In particular, the Examiner has alleged that there is no written description or support for the claimed providing each of multiple users the ability to specify a corresponding custom task associated with an extension point in a work flow "without editing said work flow".

As to the applicable standard for written description requirement, when a disclosure describes a claimed invention in a manner that permits one skilled in the art to reasonably conclude that the inventor possessed the claimed invention the written description requirement is satisfied. (MPEP §2163). This possession may be shown in any number of ways and an Applicant need not describe every claim feature exactly because there is no in *haec verba* requirement. (MPEP § 2163).

Rather, to satisfy the written description requirement, all that is required is "reasonable clarity." (MPEP § 2163.02). Also, an adequate description may be made in any way through express, implicit, or even inherent disclosures in the application, including words, structures, ***figures, diagrams***, and/or formulae. (MPEP §§ 2163(I),

2163.02). Finally, it is important to be mindful of the generally inverse correlation between the level of skill and knowledge in the art and the specificity of disclosure necessary to satisfy the written description requirement. (MPEP § 2163(II)(A)(2)) (inventions in ***"predictable" or "mature" require a lesser showing of possession than inventions in more "unpredictable" arts***).

Here, the subject matter of instant application is clearly in predictable arts since it relates to computer technology. The requisite showing is therefore less. Figure 9 clearly demonstrates work flow 920 (containing built-in tasks 910-919), ***which is not edited***, while any of a desired custom task (CT1-CT4) is associated with respective extension points (EP1-EP4).

Thus, the claimed 'not edited" feature is inherent to the disclosure of the subject application as filed. Thus, the requisite support and possession are both present in the application content, as filed.

The Honorable Board is accordingly respectfully requested to overturn the rejection under 35 U.S.C. § 112.

B. Rejection of claims 1, 3, 4, 8-11, 13-14, 18-21, 23-24, 28-31 and 32-34 under 35 U.S.C. § 103(a)

Claims 1, 3, 4, 8-11, 13-14, 18-21, 23-24, 28-31 and 32-34 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Chatterjee *et al* (US Patent # 5,774,661) in view of Hansen *et al* (US Patent #7,013,316) and Bacon *et al* (US Patent #6,430,538).

At least when the ***'not edited'*** feature (rejected based on 35 U.S.C. § 112) is properly ***considered***, each of the independent claims is clearly allowable over the art of record.

For example, with respect to Chatterjee, the Examiner had stated:

> ... Although Chatterjee does indeed relate to a workflow editor that allows a user to modify/define workflows, this does not preclude a user from defining a workflow to be

> used by others. Further, ***the user of the Chatterjee system may modify a pre-existing workflow or define a new workflow to yield a "custom" task for execution by themselves or others***. (Starting last paragraph on page 2 of Outstanding Final Office Action, ***Emphasis Added***)

It is respectfully pointed out that the modification of the above-emphasized portion requires editing of the workflow, contrary to the recitation of claim 1.

The Examiner does not rely on Bacon for this feature. However, the attention of the Honorable Board's attention is drawn to the fact that the different personal subflows of Bacon are also defined by a developer:

> ***In defining the personal subflow, the developer may specify rule-based branch conditions to specify which activity should be a next activity given a set of existing conditions.*** The given set of conditions may be specified as an expression defined according to a predefined grammar and that has work item contents data as variables in the expression. Moreover, a user interface may be provided to facilitate the developer's entry of valid expressions. The expressions so made may be used to express expert-based rules to facilitate the participant's performance of the personal subflow activity. The actual expressions are implementation dependent. (Col. 9 lines 26-38 of Bacon, ***Emphasis Added***)

Thus, though different personal subflows of Bacon may be operative as against different users/actors, the personal subflows of Bacon are specified by an individual developer defining the workflow. Therefore, the user is not provided the ability to specify the personal subflow to be executed.

In addition, claim 1 recites that, "..continuing execution of said sequence of built-in tasks ***from said extension point in said built-in task*** after executing said custom task in said instance..." (***Emphasis Added***). In other words, after completion of a custom task, execution continues after the same point in the built-in task at which execution of the custom task was started.

The Examiner relies on Figure 3 of Chatterjee for such a feature (see page 3 last paragraph of the Outstanding Final Office Action). The analogy is erroneous as

explained below.

The Honorable Board's attention is first drawn to the fact that 312 of Chatterjee is a mere decision point, and cannot be reasonably equated to the claimed extension point. In particular, the decision point of Chatterjee merely provides for a choice of one of multiple paths to be executed. In sharp contrast, as emphasized in the claim recitation above, specifying a custom task at an extension point implies that control returns to the same point upon completion of the custom task.

In that respect, the Examiner finds such a feature in the execution flow 312-313-314 and 317 of Figure 3 of Chatterjee. The Honorable Board's attention is respectfully directed to the fact that transfer of control from 314 to 317 in such a flow is due to the operation of ***decision point 314*** (not due to inherent operation by virtue of decision point 312). Accordingly, the decision point 312 of Chatterjee cannot be reasonably equated to the claimed operation of extension point.

In other words, for the above noted feature to be present in Chatterjee, control needs to return to 312 after completion of the tasks 313 and 314 in Chatterjee, without having to rely on decision point 314.

At least for some of the reasons noted above, independent claim 1 is allowable over the art of record. The remaining independent claims are also allowable over the art of record for at least some of the reasons noted above. The dependent claims are allowable over the art of record at least as depending from the allowable base claims.

Claim 4 is independently allowable in reciting that the available extension points (defined by a designer) are determined and then displayed to a user. The method further displays the custom tasks, so that the user can associate specific desired custom task with a corresponding extension point.

The Examiner relies on Figures 3 and 4 of Chatterjee for such a feature. It is respectfully pointed out that Figure 4 merely provides the details of a user interface for

defining a decision point. There is no disclosure or suggestion there that can be reasonably equated to the determining of the ***available extension points.***

In this regard, the Examiner had stated that:

> (col. 8, lines 15-41; Figures 3-4; A workflow builder display allows a user to customize a workflow by inserting decision points, where the decision points come from ***a set of predefined conditional statements***. (Page 12, second paragraph of the Outstanding Final Office Action, ***Emphasis Added***)

It is Applicant's position that the claimed available extension points cannot be reasonably equated to the predefined conditional statements. At least for such a reason, claim 4 is allowable over the art of record.

C. Rejection of claims 5-6, 15-16, 25-26, and 35-36 under 35 U.S.C. § 103(a)

Each of the claims 5-6, 15-16, 25-26, and 35-36 depends from an allowable base claim and is thus allowable over the art of record.

Conclusion

The reversal of the Examiner's rejections of claims 1, 3-6, 8-11, 13-16, 18-21, 23-26, 28-31 and 33-36 is respectfully requested. The Office is invited to telephone the Undersigned Representative at 443-552-7281 (4AM-noon EST, else voiecmail) if it is believed that an interview might be useful for any reason. The Director is hereby authorized to charge any underpayment of fees (including extension fees), or credit any overpayments to Deposit Account No.: 20-0674.

Respectfully submitted,
/Narendra Reddy Thappeta/
Signature

Date: 29-Mar-2010

Printed Name: Narendra Reddy Thappeta
Attorney for Applicant
Registration Number: 41,416

VIII. CLAIMS

Claim 1 (Previously Presented): A method of enabling a user to extend a work flow for synchronization/consolidation of data between at least two data sources, said work flow for execution in a meta directory server, said method comprising:

providing, by a designer, a sequence of built-in tasks which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources, a built-in task in said sequence of built-in tasks containing an extension point at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances,

wherein each user is provided the ability to specify a corresponding custom task associated with said extension point without editing said work flow, the custom task specified by a user containing corresponding program logic to provide a customization desired by the user,

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point;

receiving from said user data indicating a custom task associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user for the corresponding desired customization;

executing said custom task when said extension point is reached during execution of said built-in task in an instance of said work-flow for said user; and

continuing execution of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/ consolidating data between said two data sources.

Claim 2 (Canceled)

Claim 3 (Original): The method of claim 1, wherein said custom task contains an another extension point, said method further comprises receiving from said user data indicating an another custom task to be executed when said another extension point is reached during execution of said custom task.

Claim 4 (Previously Presented): The method of claim 3, further comprising:

determining a corresponding set of extension points available in each of said sequence of built-in tasks;

displaying each of said set of extension points associated with a corresponding one of said sequence of built-in tasks;

displaying said custom task and said another custom task; and

enabling said user to specify said custom task associated with said extension point, and said another custom task associated with said another extension point.

Claim 5 (Previously Presented): The method of claim 3, further comprising enabling said user to specify that said custom task is to be executed synchronously, wherein execution of said sequence of built-in tasks is suspended at said extension point during execution of said custom task, and

wherein execution o said sequence of built-in tasks is resumed after completion of execution of said custom task such that said custom task is executed in a synchronous manner.

Claim 6 (Previously Presented): The method of claim 3, further comprising enabling said user to specify that said custom task is to be executed asynchronously, wherein said custom task is executed in parallel with execution of built-in task from said extension point such that said custom task is executed in an asynchronous manner.

Claim 7 (Canceled)

Claim 8 (Previously Presented): The method of claim 1, wherein at least one of said two data sources comprises a relational database.

Claim 9 (Original): The method of claim 3, further comprising providing an utility to indicate that a specific one of said extension points is reached.

Claim 10 (Previously Presented): The method of claim 3, further comprising providing an utility in each of said sequence of built-in tasks and said custom task, wherein said utility indicates extension points available in a corresponding task.

Claim 11 (Previously Presented): A computer readable medium storing one or more sequences of instructions for causing a meta directory server to enable a user to extend a work flow for synchronization/consolidation of data between at least two data sources, wherein execution of said one or more sequences of instructions by one or more processors contained in said meta directory server causes said meta directory server to perform the actions of:

providing, by a designer, a sequence of built-in tasks which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources, a built-in task in said sequence of built-in tasks containing an extension point at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances,

wherein each user is provided the ability to specify a corresponding custom task associated with said extension point without editing said work flow, the custom task specified by a user containing corresponding program logic to provide a customization desired by the user,

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point;

receiving from said user data indicating a custom task associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user;

executing said custom task when said extension point is reached during execution of said built-in task in an instance of said work-flow for said user; and

continuing execution of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/ consolidating data between said two data sources.

Claim 12 (Canceled):

Claim 13 (Original): The meta directory server of claim 11, wherein said custom task contains an another extension point, further comprises receiving from said user data indicating an another custom task to be executed when said another extension point is reached during execution of said custom task.

Claim 14 (Previously Presented): The meta directory server of claim 13, further comprising:

determining a corresponding set of extension points available in each of said sequence of built-in tasks;

displaying each of said set of extension points associated with a corresponding one of said sequence of built-in tasks;

displaying said custom task and said another custom task; and

enabling said user to specify said custom task associated with said extension point, and said another custom task associated with said another extension point.

Claim 15 (Previously Presented): The meta directory server of claim 13, further comprising enabling said user to specify that said custom task is to be executed synchronously, wherein execution of said sequence of built-in tasks is suspended at said extension point during execution of said custom task, and

wherein execution o said sequence of built-in tasks is resumed after completion of execution of said custom task such that said custom task is executed in a synchronous manner.

Claim 16 (Previously Presented): The meta directory server of claim 13, further comprising enabling said user to specify that said custom task is to be executed asynchronously, wherein said custom task is executed in parallel with execution of built-in task from said extension point such that said custom task is executed in an asynchronous manner.

Claim 17 (Canceled)

Claim 18 (Previously Presented): The meta directory server of claim 11, wherein at least one of said two data sources comprises a relational database.

Claim 19 (Original): The meta directory server of claim 13, further comprising providing an utility to indicate that a specific one of said extension points is reached.

Claim 20 (Previously Presented): The meta directory server of claim 13, further comprising providing an utility in each of said sequence of built-in tasks and said custom task, wherein said utility indicates extension points available in a corresponding task.

Claim 21 (Previously Presented): A meta directory server enabling a user to extend a work flow for synchronization/consolidation of data between at least two data sources, said meta directory server comprising:

means for providing, by a designer, a sequence of built-in tasks which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources, a built-in task in said sequence of built-in tasks containing an extension point at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances,

wherein each user is provided the ability to specify a corresponding custom task associated with said extension point without editing said work flow, the custom task specified by a user containing corresponding program logic to provide a customization desired by the user,

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point;

means for receiving from said user data indicating a custom task associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user for the corresponding desired customization;

means for executing said custom task when said extension point is reached during execution of said built-in task; and

means for continuing execution of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/ consolidating data between said two data sources.

Claim 22 (Canceled):

Claim 23 (Original): The meta directory server of claim 21, wherein said custom task contains an another extension point, further comprises means for receiving from said

user data indicating an another custom task to be executed when said another extension point is reached during execution of said custom task.

Claim 24 (Previously Presented): The meta directory server of claim 23, further comprising:

means for determining a corresponding set of extension points available in each of said sequence of built-in tasks;

means for displaying each of said set of extension points associated with a corresponding one of said sequence of built-in tasks;

means for displaying said custom task and said another custom task; and

means for enabling said user to specify said custom task associated with said extension point, and said another custom task associated with said another extension point.

Claim 25 (Previously Presented): The meta directory server of claim 23, further comprising enabling said user to specify that said custom task is to be executed synchronously, wherein execution of said sequence of built-in tasks is suspended at said extension point during execution of said custom task, and

wherein execution o said sequence of built-in tasks is resumed after completion of execution of said custom task such that said custom task is executed in a synchronous manner.

Claim 26 (Previously Presented): The meta directory server of claim 23, further comprising enabling said user to specify that said custom task is to be executed asynchronously, wherein said custom task is executed in parallel with execution of built-in task from said extension point such that said custom task is executed in an asynchronous manner.

Claim 27 (Canceled)

Claim 28 (Previously Presented): The meta directory server of claim 21, wherein at least one of said two data sources comprises a relational database.

Claim 29 (Original): The meta directory server of claim 23, further comprising an utility means to indicate that a specific one of said extension points is reached.

Claim 30 (Previously Presented): The meta directory server of claim 23, further comprising an utility means in each of said sequence of built-in tasks and said custom task, wherein said utility means indicates extension points available in a corresponding task.

Claim 31 (Previously Presented): A meta directory server enabling a user to extend a work flow for synchronization/consolidation of data between at least two data sources, said meta directory server comprising:

a task registry block storing data related to a sequence of built-in tasks which together when executed implement said work flow for synchronization/consolidation of data between at least two data sources, a built-in task in said sequence of built-in tasks containing an extension pointpoint at a point of interest in said work flow for users,

said work flow being designed for execution by multiple users as corresponding instances,

wherein each user is provided the ability to specify a corresponding custom task associated with said extension point without editing said work flow, the custom task specified by a user containing corresponding program logic to provide a customization desired by the user,

said flow being designed to execute the specified desired custom task in the corresponding instance if specified by corresponding user at said extension point;

a user interface module receiving from said user, data indicating a custom task associated with said extension point, wherein said custom task is separate from said sequence of built-in tasks and contains a program logic specified by said user for the corresponding desired customization; and

work-flow manager module for executing said custom task when said extension point is reached during execution of said built-in task, and in addition for continuing execution of said sequence of built-in tasks from said extension point in said built-in task after executing said custom task in said instance such that all of said sequence of built-in tasks are executed to complete synchronizing/ consolidating data between said two data sources.

Claim 32 (Canceled):

Claim 33 (Original): The meta directory server of claim 31, wherein said custom task contains an another extension point, wherein said user interface further receives data

indicating an another custom task to be executed when said another extension point is reached during execution of said custom task.

Claim 34 (Previously Presented): The meta directory server of claim 33, wherein said user interface modules displays each of said set of extension points associated with a corresponding one of said sequence of built-in tasks, and enables said user to specify said custom task associated with said extension point and said another custom task associated with said another extension point.

Claim 35 (Previously Presented): The meta directory server of claim 33, wherein said user interface enables said user to specify that said custom task is to be executed synchronously, wherein execution of said sequence of built-in tasks is suspended at said extension point during execution of said custom task, and

wherein execution o said sequence of built-in tasks is resumed after completion of execution of said custom task such that said custom task is executed in a synchronous manner.

Claim 36 (Previously Presented): The meta directory server of claim 33, wherein said user interface enables said user to specify that said custom task is to be executed asynchronously, wherein said custom task is executed in parallel with execution of built-in task from said extension point such that said custom task is executed in an asynchronous manner.

IX. EVIDENCE APPENDIX

None

X. RELATED PROCEEDINGS APPENDIX

None